

組 番 氏名

1. 初期画面のプログラム解説

	コード	内容
1.	# Imports go at the top	①コメントアウト
2.	from microbit import *	②ライブラリを読み込 *はすべてのものを表す。 ※すなわち microbit コードライブラリからすべてのものを使用できるようにしてほしいという意味
3.		
4.		
5.	# Code in a 'while True:' loop repeats forever	
6.	while True:	③インデントがさがった部分をずっと行う：
7.	display.show(Image.HEART)	④LED に"HEART"というイメージを出す
8.	sleep(1000)	⑤1000 ミリ秒の間、動きを止める 1000 ミリ=1 秒
9.	display.scroll('Hello')	⑥ディスプレイに「Hello」とスクロールする

①コメントアウト 実際のプログラムの動作に影響しない文字列。

②マイクロビットのプログラムのライブラリ（プログラムコード集）を呼び出して使えるようにする。

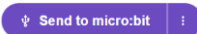
③while で始まる文は「～の間：以降の事をする」という意味。この場合はずっと：以降の動作を繰り返すということになる。

④ディスプレイにイメージを出す。この場合は、micro:bit のライブラリに収録されている「HEART」というイメージを出している。

⑤sleep()はかっこの中にある時間、プログラムの動作を止める。かっこ内の単位はミリ秒。

⑥ディスプレイにかっこの中の文字列をスクロールさせる（日本語は使用できない）。

2. 転送方法

プログラムを書いたら、micro:bit とパソコンをケーブルでつなぎ、左下の  を押す。「ペア」を押し、左上に「BBC micro:bit CMSIS-DAP」が出てくるので、それを選択して、接続ボタンを押す。（最初の転送は遅いが、二回目以降は早くなる。）

課題 1-1★ LED にスクロールさせる文字列を変えてみよう。（プログラムの画面の写真提出）
※アルファベットしか対応していません。

課題 1-2★ HEART のイメージを 10 秒間表示させてみよう。（プログラムの画面の写真提出）

課題 1-3★★ HEART 以外にも以下のイメージがライブラリに収録されている。気になるものを表示させてみよう。(micro:bit の LED 画面の写真提出)

Image.HEART	Image.HEART_SMALL	Image.HAPPY	Image.SMILE	Image.SAD
Image.CONFUSED	Image.ANGRY	Image.ASLEEP	Image.SURPRISED	Image.SILLY
Image.FABULOUS	Image.MEH	Image.YES	Image.NO	Image.CLOCK*
Image.ARROW_**	Image.TRIANGLE	Image.TRIANGLE_LEFT	Image.CHESSBOARD	Image.DIAMOND
Image.DIAMOND_SMALL	Image.SQUARE	Image.SQUARE_SMALL	Image.RABBIT	Image.COW
Image.MUSIC_CROTCHET	Image.MUSIC_QUAVER	Image.MUSIC_QUAVERS	Image.PITCHFORK	Image.XMAS
Image.PACMAN	Image.TARGET	Image.TSHIRT	Image.ROLLERSKATE	Image.DUCK
Image.HOUSE	Image.TORTOISE	Image.BUTTERFLY	Image.STICKFIGURE	Image.GHOST
Image.SWORD	Image.GIRAFFE	Image.SKULL	Image.UMBRELLA	Image.SNAKE

課題 1-4★★★★ これまでの知識を使い、3 回連続同じイメージが点滅する方法を考えてみよう。点滅したあと、Hello を表示させよう。(プログラムの画面の写真提出)

ヒント：♥ 何もない文字 ♥ 何もない文字 ♥ 何もない文字 Hello

※もっとエレガントな方法がありますが、今回はこのように考えてみてください。

組 番 氏名

プログラムを書いていると、どこまではうまくいって、どこからうまくいかないのかを切り分けるために、コメントアウトをよく使います。

コメントアウト, アンコメント Ctrl + / ※iPad の場合は Command + /
元に戻りたい場合 Ctrl + Z ※iPad の場合は Command + Z

※複数行の場合、カーソルの位置から Shift を押しながら ↓ を押して複数行を選択して Ctrl + /

コード	解説
<pre> 1 # Imports go at the top 2 from microbit import * 3 4 5 a = 5 6 b = 15 7 c = '13' 8 d = '40' 9 10 e = a + b 11 e = a - b 12 e = a * b 13 e = a / b 14 e = a + c 15 e = c + d 16 17 while True: 18 display.scroll(e)</pre>	変数 a に 5 という値を代入する。 変数 b に 15 という値を代入する。 変数 c に 13 という文字列を代入する。 変数 d に 40 という文字列を代入する。 エラーになりますがとりあえず書いてください。 T は大文字、コロンを忘れずに。 半角スペース 4 つ分あけて、このように書く。

課題 2-1★ 次の①～⑨の課題を行い、マイクロビットに書かれている表示を記入してみよう。エラーが出る場合、何エラーなのかも書いてください。(このプリントの写真提出)

①まずは 10 行目から 15 行目までコメントアウトしてください。

②10 行目だけアンコメントし実行した際に、どんな表示になるか。 表示 _____

③11 行目だけアンコメントし実行した際に、どんな表示になるか。 表示 _____

④12 行目だけアンコメントし実行した際に、どんな表示になるか。 表示 _____

⑤13 行目だけアンコメントし実行した際に、どんな表示になるか。 表示 _____

⑥14 行目だけアンコメントし実行した際に、どんな表示になるか。 表示 _____

⑦15 行目だけアンコメントし実行した際に、どんな表示になるか。表示_____

⑧10～15 行目をアンコメントし実行した際に、どんな表示になるか。表示_____

(すべての行を実行)

課題 2-2 わかったこと★★ 少なくとも三つは記述 (このプリントの写真提出)

- ・ 割り算の答えのこと。 ・ 文字列と数値の足し算のこと ・ 文字列と文字列の足し算のこと。
- ・ エラーのコードがあった際に、表示されるメッセージのこと。

課題 2-3★★ 上のプログラムの部分を一部消すか、コメントアウトするかして、次の①～②の内容のプログラムを書いて、答えをプリントに書いてください。(このプリントの写真提出)

①変数 a に 5 を入れて、a を 0 で割った答えを表示させる。表示_____

②変数 a に 2 を入れて、それを 16 乗した値を表示させる。表示_____

ヒント べき乗は**で表現することができます。

課題 2-4★★★★ 変数 x に 7、変数 y に 3 を代入し、その変数同士の足し算の答えを z という変数に入れ、表示で「No10」となるようなプログラムを書いてください。(プログラムの写真提出)

ヒント

- ① 計算結果を整数値ではなく、文字列の型にしたい場合、`z = str(z)`とすれば、文字列に変換されます。
- ② それを 2-1 の 15 行目のような処理をすれば表示できます。

課題 2-5★★★★ テスト対策部分に出したエラー以外にも様々なエラーがある。どのようなエラーがあるか調べてみよう。「Python エラー 一覧」などと調べてみましょう。内容がわからなくても、エラー名はわかったりするので、エラー名とその説明を書いてください。(プリントの写真提出)

エラー名	説明
Error	
Error	
Error	

【テスト対策部分】

1. コメントアウト, アンコメント

プログラムの行の先頭に#ハッシュマークをつけてプログラムの実行を無効化することを（ ）という。コメントアウトしたい行を選択する方法は 2 通りあり、①マウスで選択する方法、②（ ）を押しながら、（ ）を押すことで選択することができる。選択したあとで、（ ）を押すことで、コメントアウトができ、再び押すと#をとる（ ）ができる。コメントアウトやアンコメントを利用することで、効率的にプログラムを書くことができる。もし、あるコードを消してしまうと、再度必要になったときに再び記述しなければならないので、効率が悪い。コメントアウト, アンコメントを使う利点は今までに書いた（ ）点や、（ ）ことができる点である。

	コード	実行される行
1.	コード①	→ →
2.	コード②	
3.	# コード③	
4.	# コード④	
5.	コード⑤	

	コード	実行される行
1.	# コード①	→ →
2.	コード②	
3.	# コード③	
4.	コード④	
5.	コード⑤	

※似たようなコードを 2 行書いておき、コメントアウトでそれぞれを切り替えするなどの操作を覚えておくと楽です。

2. 変数

変数は（ ）や（ ）を保管する箱のようなもので、代入演算子（ ）を使って最初に変数に値を代入した時点で変数の宣言が行われる。たとえば $a = 2$ とすれば、変数 a に 2 が代入される。 $2 = a$ という表記はできない点に注意する。※近年では、変数を箱ではなくラベル（名札や付箋）として捉え、値や文字列にラベルを付けるという概念が登場してきている。

使い方

①代入・・・文字列や数値を変数に入れる。

コード	意味
$a = 3$ $b = '3'$ $c = 'Hello'$	変数 a に 3 という値を代入する。 変数 b に 3 という文字列を代入する。'でも"でも可。 変数 c に Hello という文字列を代入する。

②上書き・・・変数に値や文字列が入っている状態から再度代入して上書きする。

コード	意味
$a = 10$ $a = 20$ $display.scroll(a)$	変数 a に 10 という値を代入する。 変数 a に 20 という値を代入（上書き）する。 すると 20 という数字が表示される。

③入れ替え（変数 a と変数 b を入れ替えたい場合）※共通テストにもよくでてくると思います。

コード	意味
$a = 10$ $b = 20$ $temp = a$ $a = b$ $b = temp$	変数 a に 10 という値を代入する。 変数 b に 20 という値を代入する。 $temp$ （一時的なという意味）変数に変数 a を代入し、 変数 b には変数 a の値を代入し、 変数 b は $temp$ 変数の値を代入する。

④（変数の発展版の）オブジェクト変数

これまで a や b などに入れていたものは値や文字列であるが、Python では設計図から作られた実体を入れることもでき、この場合、オブジェクト変数という。

```
motor = WheelMotor()
```

などと書くが、これはまた後程説明していく。

3. 演算子一覧

次は Python でよく使われる演算子の例である。

算術・結合演算子		比較演算子		代入演算子	
+	数値の足し算 ()	==	()	=	()
		!=	()	論理演算子	
-	引き算	<	未満（より小さい）	and	かつ
*	()	>	超過（より大きい）	or	または
/	()	()	以下	not	ではない
%	()	()	以上	ドット演算子	
//	()			.	～の
**	()				

例： a = 13 // 3 とすれば、a に 4 が代入され、 b = 2 ** 3 とすれば、b に 8 が代入される。

4. エラーの種類

プログラムを書いているとエラーがたくさん出てきます。把握しておきましょう。

①.(ドット)のあとの単語が間違っている可能性がある。	()
②特殊記号が間違っている可能性がある。	()
③変数や.(ドット)の前の単語が間違っている可能性がある。	()
④字下げ（インデント）が適切に行われていない可能性がある。	()
⑤モジュールが適切に読み込まれていない可能性がある。	()
⑥文法上も問題なく実行も途中で止まることもないが、意図した結果が得られない場合のエラー	()

組 番 氏名

ゲームなどでランダムな数字を発生させ、その値に応じてガチャのアイテムを引かせたりする場合があります。どのように実装しているのでしょうか？

	コード	解説
1.	# Imports go at the top	今回はここが追加されることに注意！ インデント下がった部分をずっと行う。 randint はランダムの整数値を返し、変数 value に代入。この場合 1～3 まで。int は INTEGER の略。 変数 value に入った値を表示させる。
2.	from microbit import *	
3.	import random	
4.		
5.	while True:	
6.	value = random.randint(1,3)	
7.	display.scroll(value)	
8.		

課題 3-1★ 上記のプログラムでランダムが生成されるかどうか調べる。(プログラムの画面の写真提出)

課題 3-2★ 乱数が 1 のときは gu, 2 のときは cho, 3 のときは pa と表示させるためのプログラムは以下のように書く。(プログラムの画面の写真提出)

	コード	解説
1.	# Imports go at the top	この行をコメントアウトし、以下を書いていこう。 :を入力してエンターキーを押すと、 半角スペース 4 つ分のあとにカーソルが現れる。 バックスペースを 1 回押すと 4 つ左に移動。 最後は else:でよい
2.	from microbit import *	
3.	import random	
4.		
5.	while True:	
6.	value = random.randint(1,3)	
7.	# display.scroll(value)	
8.	if value <= 1:	
9.	display.scroll('gu')	
10.	elif value <= 2:	
11.	display.scroll('cho')	
12.	else:	
13.	display.scroll('pa')	

このようにすると、gu や cho や pa ではなく、手のマークにしたいくなります。その場合、gu や cho や pa というオブジェクト変数に、Image を使ってマークを作り登録し、それを呼び出していくようなプログラムを書いています。

課題 3-3★★ gu のイメージを作り、表示させよう。(プログラムの画面の写真提出)

	コード	解説
1.	# Imports go at the top	
2.	from microbit import *	

3.	import random	
4.		
5.	gu = Image("00000:"	"00000:"を作り，コピー(Ctrl + C)してエンターキーを押す。
6.	"00000:"	貼り付け(Ctrl + V)をする。
7.	"09999:"	
8.	"99999:"	
9.	"99999")	最後の行はコロンがないことに注意。コロンを削除する。
10.		あとは gu のマークになるように適宜書き換える。
11.	while True:	
12.	display.show(gu)	display.scroll ではなく，display.show(変数名)を使う。
13.	sleep(2000)	その変数の最低表示時間を設定する。今回は 2 秒

課題 3-4★★ gu,cho,pa の手のマークを作り，乱数によって表示が異なるプログラムを作ろう。(プログラムの画面の写真全行提出)

	コード	解説
1.	# Imports go at the top	
2.	from microbit import *	
3.	import random	
4.		
5.	gu = Image("00000:"	
6.	"00000:"	
7.	"09999:"	
8.	"99999:"	
9.	"99999")	
10.		
11.	cho = Image("09090:"	gu の 5 行分をコピー&ペーストし，適宜編集し直す。
12.	"09090:"	
13.	"09090:"	
14.	"09999:"	
15.	"99999")	
16.		
17.	pa = Image("90909:"	gu の 5 行分をコピー&ペーストし，適宜編集し直す。
18.	"90909:"	エイリアンみたいな手の指になりますが，気にしないでください。
19.	"90909:"	
20.	"99999:"	
21.	"99999")	
22.		
23.	while True:	
24.	value = random.randint(1,3)	
25.	if value <= 1:	
26.	display.show(gu)	
27.	elif value <= 2:	
28.	display.show(cho)	
29.	else:	
30.	display.show(pa)	
31.	sleep(2000)	gu や cho や pa どれでも 2 秒停止させる。インデントはこの位置にあることに注意！一番左から半角スペース 4 つ分

課題 3-5★★★★ (LED 自作)LED に表示させるマークを自分で定義し、それを表示させるプログラムを書いてください。(micorbit の LED の動画提出)

※もし点滅表示させたい場合は以下のコードを参考にしてください。

例：gu というマークを点滅させたい場合、

```
while True:
    display.show(gu)
    sleep(500)
    display.clear()
    sleep(500)
```

と書くと、0.5 秒間隔で点滅が起こります。

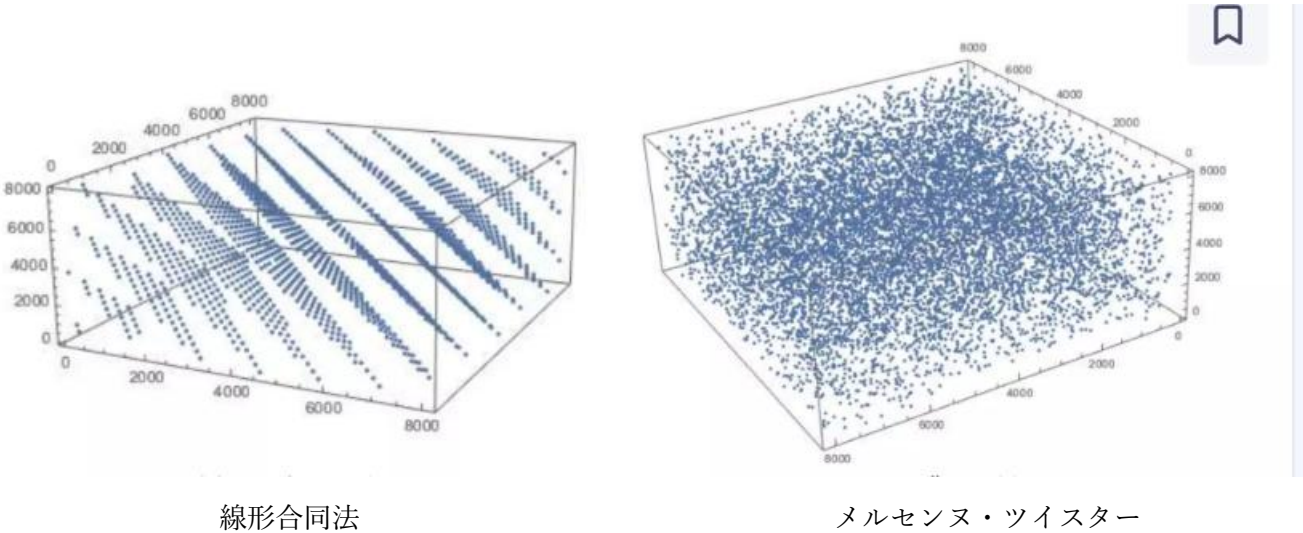
【テスト対策部分】

1. 疑似乱数

乱数列のように見えるが、実際には確定的な計算によって求めている擬似乱数列による乱数

- ・ () …昔の疑似乱数生成アルゴリズム。ランダム性がないと指摘されている。
- ・ () …現在主流の疑似乱数生成アルゴリズム。日本人が考案。

micro:bit の場合、疑似乱数を扱いたい場合、3 行目に () と書く。



2. IF 文のまとめ

何か値を判別したり、「押す・押さない」の判別をしたりするなど、条件に応じた挙動を実行させるためには、IF 文を使う。IF 文は指定した条件式が真の時にだけ処理を実行。

IF 文の基本

1 通りの場合	2 通りの場合	3 通り以上の場合
if 条件式 1: 処理 A	if 条件式 1: 処理 A else: 処理 B	if 条件式 1: 処理 A elif 条件式 2: 処理 B else: 処理 C

※コロンの(:)を入力してエンターボタンを押すと、半角スペース 4 つ分右にずれる。

例：100 点満点の点数を score という変数に入れて判別したい場合。

分類	1 通りの場合	2 通りの場合	3 通り（以上）の場合
コード	<pre> display.scroll('Akaten') </pre>	<pre> display.scroll('Akaten') display.scroll('Gokaku') </pre>	<pre> display.scroll('Akaten') display.scroll('Normal') display.scroll('Excellent') </pre>
考え方	〇〇未満（以下，以上，より大きい，と等しい）か。	〇〇未満（以下，以上，より大きい，と等しい）か， それ以外 か。	40 点未満でまずは判別。次に 75 点未満で判別し，あとはそれ以外。

同じことだけど・・・？

別に以下のように書いたって動くけど・・・	以下のように書く方がエレガント
<pre> if score < 40: display.scroll('Akaten') if score >= 40: display.scroll('Gokaku') </pre>	<pre> if score < 40: display.scroll('Akaten') else: display.scroll('Gokaku') </pre>

別に以下のように書いたって動くけど・・・	以下のように書く方がエレガント
<pre> if score < 40: display.scroll('Akaten') if score >= 40 and score < 75: display.scroll('Normal') if score >= 75: display.scroll('Excellent') </pre>	<pre> if score < 40: display.scroll('Akaten') elif score < 75: display.scroll('Normal') else: display.scroll('Excellent') </pre>

比較演算子一覧

比較演算子		比較演算子	
()	等しい	()	超過（より大きい）
()	等しくない	()	以下
()	未満（より小さい）	()	以上

※=一つであると代入を意味するので，イコールは==と記入。

組 番 氏名

IF 文はよく使うので繰り返し学習していきます。ボタンの使い方 IF 文です。

じゃんけんプログラム…a ボタンを押したら、「じゃんけん・・・(手)」と表示し、b ボタンがおされたら、「あいこで・・・(手)」と表示させる。

コード	解説
<pre> 1. # Imports go at the top 2. from microbit import * 3. import random 4. 5. gu = Image("00000:" 6. "00000:" 7. "09999:" 8. "99999:" 9. "99999") 10. 11. cho = Image("09090:" 12. "09090:" 13. "09090:" 14. "09999:" 15. "99999") 16. 17. pa = Image("90909:" 18. "90909:" 19. "90909:" 20. "99999:" 21. "99999") 22. 23. while True: 24. x = random.randint(1,3) 25. if button_a.was_pressed(): 26. #display.scroll(x) 27. display.scroll("janken") 28. if x <= 1: 29. display.show(gu) 30. elif x <= 2: 31. display.show(cho) 32. else: 33. display.show(pa) 34. if button_b.is_pressed(): 35. display.scroll("aikode") 36. if x <= 1: 37. display.show(gu) </pre>	ランダムオブジェクトを使うためランダムを import。 じゃんけん変数を設定しておく。 今まで"gu"と書いていたが、gu という変数を扱いたい場合はそのまま gu と書く。 一回だけ実行される部分 ここに今後使う変数を書いていく。 乱数を発生させる ボタンが押されたという if 文。ここは was で設定 if 文の中にさらに if 文を書くことも可能。 gu という変数名を扱いたい場合は gu gu と表示させたい場合は"gu" 33 行目まで書いたら下は書かずに一回実行してみる。 ここは is で設定してみる。上をコピーし、a.was→b.is にかえる。if button_b.is_pressed():の位置に注意！ 何回も繰り返される部分。

インデント (タブ) あるいはスペース 4

38.	elif x <= 2:	
39.	display.show(cho)	
40.	else:	
41.	display.show(pa)	

課題 4-1★ 上記のプログラムを書く。(プログラムの写真全行提出)

課題 4-2★★ 以下はランダムに数字を発生させ、その結果によってじゃんけんの文字をスクロールさせるプログラムである。以下のプログラムで間違いの部分はどこか答えなさい。(プリントの写真提出)

	コード	解説
1.	# Imports go at the top	このプログラムでは、5 行目で乱数 x の値が確定してしまい、その後、乱数 x の値がずっと使われる。 (A.) 行目のコードを (B.) 行目と (C.) 行目の間 または、 (D.) 行目と (E.) 行目の間 または、 (F.) 行目と (G.) 行目の間 のどこかにもっていかなければいけない。
2.	from microbit import *	
3.	import random	
4.		
5.	x = random.randint(1,3)	
6.	while True:	
7.	if button_a.was_pressed():	
8.	display.scroll("janken")	
9.	if x <= 1:	
10.	display.scroll("gu")	
11.	elif x <= 2:	
12.	display.scroll("cho")	
13.	else:	
14.	display.scroll("pa")	

課題 4-3★★a ボタンが押されたら、さいころ 1 個をふった目（・など）を表示させよう。変数名は先頭に関してアルファベットしか使えません。one,two,などの変数名を設定しよう。display.show を使います。(LED の動画提出)

課題 4-4★★★★a ボタンが押されたら、さいころ 1 個をふった目（・など）を表示させ、b ボタンが押されたらさいころ 2 個ふった目の数（2～12）を表示させよう。また、さいころ 2 個振った場合は同様に確からしい状態（random.randint(2,12)ではない。）で表示させること。(プログラムの写真提出。すべての行を撮って提出すること。途中まで書いても部分点をあげます。)

ヒント 1：さいころ 1 個の値を変数 x,y にそれぞれ入れて、足した値をさらに変数 z などに代入し、変数 z をつくること。 if z <=9: display.show(z)	ヒント 2：1～9 までは IF を使って、display.show でいけるが、10,11,12 は別の方法で表示させる必要がある。LED の表示を 10,11,12 は自作し、IF 文などを使って表示させていこう。 ヒント 2 の例 10 の場合の LED の例 <pre>ten=Image("90999:") "90909:" "90909:" "90909:" "90999")</pre>
---	---

【テスト対策部分】

1. IF 文の使い方は様々です。以下に色々な例を載せておきます。

①IF 文の不等号の使い方

IF 文は数学的な考え方を使えば、書き方は様々です。

age = ○○ #○○には年齢が入る

書き方 1

```
if age < 18:
    display.scroll('未成年')
else:
    display.scroll('成年')
```

⇒
この書き方
でも可能

書き方 2

```
if age _____ 18:
    display.scroll('_____')
else:
    display.scroll('_____')
```

不等号を変えれば同じになります。

②IF 文のネスト

IF 文の中にさらに IF 文を使うことも可能です。

書き方 1

```
if age _____ 18:
    display.scroll('未成年')
else:
    if age _____ 20:
        display.scroll('成年で飲酒 OK。')
    else:
        display.scroll('成年だが飲酒 NG。')
```

⇒
この書き方
でも可能

書き方 2

```
if age _____ 18:
    if age _____ 20:
        display.scroll('成年だが飲酒 NG。')
    else:
        display.scroll('成年で飲酒 OK。')
else:
    display.scroll('未成年')
```

③IF 文のネストを使わず、3 パターンあるという意味で考える。

②のように書かず、以下のように書いても同じ意味です。

書き方 1

```
if age _____ 18:
    display.scroll('未成年')
_____ age _____ 20:
    display.scroll('成年だが飲酒 NG。')
else:
    display.scroll('成年で飲酒 OK。')
```

⇒
この書き方
でも可能

書き方 2

```
if age _____ 20:
    display.scroll('成年で飲酒 OK。')
_____ age _____ 18:
    display.scroll('成年だが飲酒 NG。')
else:
    display.scroll('未成年')
```

3. 効率のよい書き方とインデントの位置（前回の復習）

例えば、以下のように書いた場合、gu,cho,pa のマークが 2 秒おきに変わっていきます。

	コード	解説
1.	# Imports go at the top	グーのマークを定義していく。
2.	from microbit import *	
3.	import random	
4.		
5.	gu = Image("00000:"	
6.	"00000:"	

省略

21.	"99999:"	
22.	"99999")	
23.		以下のことをずっと行う 1 ～ 3 の乱数を発生させ、value という変数に入れる。 value が 1 以下のとき（実質 1 のとき） gu を表示させ 2 秒停止させる（2 秒 gu を表示させる） もしくは value が 1 以下のとき（実質 2 のとき） cho を表示させ 2 秒停止させる（2 秒 cho を表示させる） それ以外のとき（実質 3 のとき） pa を表示させ 2 秒停止させる（2 秒 pa を表示させる）
24.	while True:	
25.	value = random.randint(1,3)	
26.	if value <= 1:	
27.	display.show(gu)	
28.	sleep(2000)	
29.	elif value <= 2:	
30.	display.show(cho)	
31.	sleep(2000)	
32.	else:	
33.	display.show(pa)	
34.	sleep(2000)	

しかし、sleep(2000)と毎回かくのはちょっと効率的ではありません。そこで次のように書きます。

24.	while True:	if 文はこの範囲で行われるという解釈をしましょう。 ここはどの if 文を選択したとしてもそのあと必ず実行される行
25.	value = random.randint(1,3)	
26.	if value <= 1:	
27.	display.show(gu)	
28.	elif value <= 2:	
29.	display.show(cho)	
30.	else:	
31.	display.show(pa)	
32.	sleep(2000)	

つまり、インデントの位置はプログラムを読み解くためには大切な理解です。

では、次の場合はどうでしょう？

24.	while True:
25.	value = random.randint(1,3)
26.	if value <= 1:
27.	display.show(gu)
28.	elif value <= 2:
29.	display.show(cho)
30.	else:
31.	display.show(pa)
32.	sleep(2000)

Q.左のプログラムではパーしか表示されない。なぜでしょうか？

- ア. 28 行目の elif は if とかかなければならないから
- イ. show には sleep が必要であるから
- ウ. 26 行目や 28 行目の <= の比較演算子がおかしいから
- エ. gu や cho は高速で表示されているが高速なため見えず、pa だけが 2 秒表示されるようになるので、pa だけが見えるようにプログラムが組まれているから

答え_____

組 番 氏名

ループに入った時のプログラムとループに入る前のプログラムでコードをどこに記述していけばよいのか考えてみよう。

タイマー 10（インデントに注意しながら入力しよう）

	コード	
1.	# Imports go at the top	
2.	from microbit import *	
3.	import music	今回はここが入ることに注意。
4.		
5.	count = 0	ループに入る前に count 変数を 0 にしておく
6.	while True:	（これを初期化という。）
7.	if button_a.was_pressed():	a ボタンが押された時
8.	count = count + 10	10 ずつ増やしていく
9.	display.scroll(count)	
10.	if button_b.was_pressed():	b ボタンが押された時
11.	while True:	ずっと繰り返す。
12.	if count == 0:	0 なら終了させる。
13.	music.play(music.RINGTONE)	音楽を鳴らす。
14.	break	ループを抜ける。
15.	display.scroll(count)	値を表示させ、
16.	count = count - 1	1 を引く。count -= 1 という書き方でもよい。
17.		

count 変数がループに入る前にあることに注意。最初に 0 にしておく。初期化という。while の中にあると、ループする度に 0 になってしまい、プログラムとして成り立たなくなる。

課題 5-1★ 上記のプログラムを書く。（プログラムの画面の写真提出）

課題 5-2★★ 次のコードは上の 12～14 行目と 15～16 行目を入れ替えただけですが、プログラムのバグなプログラムです。

	コード	
1.	# Imports go at the top	
2.	from microbit import *	
3.	import music	
4.		
5.	count = 0	
6.	while True:	
7.	if button_a.was_pressed():	
8.	count = count + 10	
9.	display.scroll(count)	
10.	if button_b.was_pressed():	書き始める位置に注意！
11.	while True:	
12.	display.scroll(count)	
13.	count = count - 1	count -= 1 という書き方でもよい。
14.	if count == 0:	
15.	music.play(music.RINGTONE)	
16.	break	11 行目のループを終わらせるための break
17.		

なぜなのか理由を書いてみよう。（このプリントの写真提出）

検証方法：プログラムを転送したあと、a ボタンを押さずに b ボタンから押してみる。

課題 5-3★★ （人数チェッカーの制作）a ボタンを押したら、押した回数をカウントし表示させ、b ボタンが押されたら、その値を表示させる。a ボタンと b ボタンが同時に押されたら、リセットさせる。

ヒント:a と b の同時押しは、if button_a.is_pressed() and button_b.is_pressed(): です。同時押しは、a ボタンを押しながら、b ボタンを押すと実現できます。リセットは本当にリセットされたかわからないので、直前で音楽を鳴らしたり、直前で display.scroll(“reset”)としたりして、わかりやすくしよう。（プログラムの写真全行提出）

ヒント：リセットとは人数を数える変数（count 変数など）をいくつにすればよいか考えてみよう。

課題 5-4★★★★ 以下のような表示になるためにはどのようにすればよいか、プログラムをインターネットで調べながらループ while を使って書きなさい。（プログラムの画面の写真提出）

表示	ヒント
1 12 123 1234 12345	・文字列を入れる変数を文字列型で初期化しておく。 ・数字を while 文で 1 ずつあげていく。 ・あげた数字を文字列にして、文字列の変数に結合させる。（文字列の結語は文字列同士の+でいけます。）

【テスト対策部分】

1. ループ（反復処理）

今までなぜかずっと素通りしてきた while True:。この部分を本日は掘り下げていく。反復処理（ループ）は while 文の他に for 文というものもあるが、これは次回行う。

while True: …「以下のことをずっと行う」という意味である。()ともいう。()をかけばループから抜けることができる。プログラムは入力を受け続けていることが多いので、基本的に while True のような無限ループをしていることが多い。

2. 初期化

コンピュータプログラムでは、プログラム内で利用するデータ、変数などを利用可能な初期状態にすることを () という。宣言した変数に初期値を書き込む処理などが該当し、プログラムの冒頭や関数、メソッドの冒頭などで実行されることが多い。変数の初期化が必要な理由は、もしも初期化をしない状態でその変数を用いて演算処理を行おうとすると、()が発生し、プログラムがとまってしまう。

例：最初に変数に代入される時に使う。

例：数値型	例：文字列型
a = 1 count = 0 など、最初に数値を代入しておく。	s = 'aaaaa' s1 = " 何か文字列を代入したり、シングルクオーテーション（あるいはダブルクオーテーション）でくくり、文字列型として定義しておく。

3. ループと初期化

初期化とループを使いプログラムを書いていくことはよく起こる。

例：a を 3 回表示させる。プログラム

コード	解説
<pre>_____ while _____ display.scroll('a') _____</pre>	初期化する。 i が 3 未満の時繰り返す。 a を表示する。 i を 1 プラスさせて、ループの最初に戻る。

これを見ると i が 0 の時 a が 1 回、1 の時 a が 1 回、2 の時 a が 1 回表示されるので、合計 a が 3 回表示される。
※ここで、i = i + 1 のように変数の値を 1 増やすことを () といい、逆に i = i - 1 のように変数の値を 1 減らすことを () という。

組 番 氏名

前回まで while 文を扱ったので今回は for 文を扱います。

	コード	解説
1.	# Imports go at the top	
2.	from microbit import *	
3.		
4.		
5.	while True:	
6.	if button_a.was_pressed():	a ボタンが押されたら,
7.	for i in range(1,11):	i を 1 から始めて 10 回数値をあげていく。
8.	display.scroll(i)	i を表示。
9.	if i % 2 == 1:	もし 2 で割って 1 余れば (奇数なら)
10.	display.show(Image.HAPPY)	ハッピーを
11.	sleep(3000)	3 秒表示させ,
12.	else:	そうでなければ,
13.	display.show(Image.ANGRY)	アングリーを
14.	sleep(3000)	3 秒表示させる。

課題 6-1★ a ボタンがおされたら, 1HAPPY 2ANGRY 3HAPPY 4ANGRY・・・と 10 まで表示さる。
(プログラムの画面の写真提出)

課題 6-2★★ b ボタンがおされたら, 10 回, 1~10 の間で乱数を発生させその数字を表示し, もし, 乱数発生時に 10 がでれば音楽を鳴らして, ループを止める。(プログラムの画面の写真提出)

ヒント 1 : import random と import music を忘れずに書くこと。

ヒント 2 : 乱数 random.randint(1,10)を発生させ, 適当な変数名に代入し, それを表示する。

ヒント 3 : もし, その変数が 10 であつたら, 音楽 music.play(music.RINGTONE)を鳴らして break する。

	コードのヒント	解説
1.	# Imports go at the top	
2.	from microbit import *	
3.	import 	
4.	import 	
5.	while True:	
6.	if button_b.was_pressed():	
7.	for i in range():	10 回繰り返す
8.	j = random.randint(1,10)	1 から 10 までの乱数を発生させる。
9.	display.scroll(j)	その数を表示する。
10.		等しいを表現する比較演算子は何だったでしょうか?
11.	music.play(music.RINGTONE)	
12.		繰り返しを抜けるためには何と書きますか?
13.		

課題 6-3★★ Num というリストに 5,7,9,11,35 を入れ、それを順番に表示させる。(プログラムの画面の写真提出)

	コード	解説
1.	# Imports go at the top	ここで Num[0]から Num[4]まで値を代入する 0 から 5 回繰り返す。 要素番号を表示させる。挙動がわかったらコメントアウト Num[0]～Num[4]までの 5 つ分を表示させる。
2.	from microbit import *	
3.		
4.	Num = [5, 7, 9, 11, 35]	
5.	for i in range(0,5):	
6.	display.scroll(i)	
7.	display.scroll(Num[i])	
8.		

課題 6-4★★ リストに入っている数字をまずは表示させ、2 からその数-1 まで表示させるプログラムを作る。

例：5,2,3,4, 7, 2,3,4,5,6, 9,2,3,4・・・。(プログラムの画面の写真提出)

ヒント 1：for 文を 2 回使います。

ヒント 2：最初の for 文の中で、配列に入っている数を表示させ、2 番目の for 文で 2 からその数-1 まで順番に表示させるプログラムを作ります。

参考までに 5 の段までの九九のプログラムを載せておきます。

	コード	解説
1.	# Imports go at the top	数値型の変数は str 関数を使って文字 列型に変換してから結合していく
2.	from microbit import *	
3.		
4.	for i in range(1,6):	
5.	display.scroll(str(i)+ "nodan")	
6.	for j in range(1,10):	
7.	display.scroll(str(i) + "*" + str(j) + "=" + str(i * j))	
8.		

※これを完成させられるかが 6-5 の課題を攻略するヒントです。

課題 6-5★★★★ 6-4 の配列の要素に対して、それぞれ素数(prime Number)か、合成数(composite Number)かを判定するプログラムを作る。(プログラムの画面の写真全行提出。途中でも部分点あげます。)

アウトプット例：「5 = p 7 = p 9 = c 11 = p 35 = c」と出力

ヒント 1：判別したい数字を解析する前に prime 変数を初期化しておく。 prime = True など

※prime = True や prime = False のように、真偽値を入れる型のことを bool 型という。

ヒント 2：for 文で操る変数 j を用意し、2 から順番にその数字-1 の数まで割っていき、一つでも割れるものがあるれば、prime = False にし、ループを抜ける（解析）。割ったあまりが 0 になるのは、if Num[i]%j == 0: などとすると表現できる。

ヒント 3：prime が True であれば「数=p」と表示し、そうでなければ「数=c」と表示する。

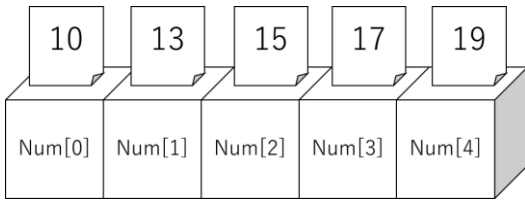
※bool 型の場合、if prime == True: という書き方でもよいし、if prime: という書き方でもよい。

ヒント 4：数値と文字列の結合は、display.scroll(str(数字の変数) + '=p')でいけます。

【テスト対策部分】

1. 配列（リスト）について

変数は一つの値や文字列しか入れることができませんが、配列は複数の値を入れて管理することができます。また、この複数の値は追加したり、削除したりすることも可能で、多くのデータを用いてプログラミングすることが可能になってきます。共通テスト用のプログラミング表記では配列は大文字から始まるので、資料も大文字から書いていますが、小文字で書いても構いません。配列は添え字が付けられ、0 から始まるのが特徴です。

コード	イメージ	呼び出す方法のコード
Num = [10,13,15,17,19]		display.scroll(Num[0]) → _____ を表示 display.scroll(Num[1]) display.scroll(Num[2]) → _____ を表示

このように添え字を指定すれば値を取り出すことが可能であります。この数が増えると書いていくことが大変になるので、for 文などが使われます。

2. for 文

今回はもう一つの反復処理である for 文を学びます。

	書き方	具体的なコード	解説
①	for 変数 in range(回数): 処理 A	for i in range(____): display.scroll('a')	a を 10 回表示
②	for 変数 in range(値 A,値 B,[増減値]): 処理 A	for a in range(____, ____): display.scroll(a)	第三引数を省略すると、+1 増える。1 ~ 10[(値 B)-1]まで表示
③	for 変数 in range(値 A,値 B,[増減値]): 処理 A	Num = [10,13,15,17,19] for i in range(____, ____): display.scroll(Num[i])	Num という配列に 10,13,15,17,19 を代入。この場合、Num[0] ~ Num[4]の値を 5 つ表示させる。

反復処理を使う場合、繰り返す回数として使いたいのか、変数の値として使いたいのか、配列の添字の番号として使いたいのかどれで使いたいかを考えながら使っていくと理解しやすくなります。

3. for 文の中に for 文

for 文の中に for 文を書くことも可能です。

例えば、九九で 1 の段から 5 の段までを表示させるプログラムは以下の通りです。

	コード	解説
1. 2. 3. 4. 5. 6. 7. 8.	<pre># Imports go at the top from microbit import * for i in range(1,6): display.scroll(str(i) + "nodan") for j in range(1,10): display.scroll(str(i) + (" ") + str(j) + "=" + str(i * j))</pre>	数値型の変数は文字列と結合できないので、str 関数を入れて、文字列型に変化させてから+で結合していく。

表示→ 1nodan 1*1=1 1*2=2 1*3=3 . . . 1*9=9 2nodan 2*1=1 2*2=4 . . . 5*9=45

4. 練習問題 1

① for a in range(10): display.scroll('b')	② for a in range(0,10): display.scroll(a)
③ for a in range(10): display.scroll('a')	④ List = [1,2,3,4,5] for i in range(0,5): display.scroll(List[i])
⑤ List = [1,2,3,4,5] for i in range(1,5): display.scroll(List[i])	⑥ Array = [2,3,4,5,6,7,8] i = 5 for j in range(2,Array[i]): display.scroll(j)

語群

ア. a を 10 回表示	イ. b を 10 回表示	ウ. 1～10 まで表示。
エ. 1～5 まで表示	オ. 0～10 まで表示	カ. 0～9 まで表示
キ. 0～4 まで表示	ク. 2～5 まで表示	ケ. 2～6 まで表示
コ. 2～7 まで表示	サ. 2～8 まで表示	

答え

① () ② () ③ () ④ () ⑤ () ⑥ ()

5. 練習問題 2

for 文の引数には変数や配列変数を指定することもできる。

次のプログラムの 1 に以下のコードを書いた場合、どのような表示になるか、記号で答えなさい。

	コード	解説
1. 2. 3. 4. 5. 6. 7. 8.	<pre># Imports go at the top from microbit import * a = 3 Num = [5, 6, 7, 8] for i in range(a,Num[a]): 1</pre>	変数 a に 3 を代入。 配列変数 Num に 5,6,7,8 を代入 i を 3 から Num[3] の 8 まで繰り返す。

空欄 1 が display.scroll(a) のとき 表示は ()

空欄 1 が display.scroll(i) のとき 表示は ()

空欄 1 が display.scroll(Num[a]) のとき 表示は ()

語群

ア. 3	イ. 3 3 3 3 3	ウ. 7 7 7 7 7
エ. 8 8 8 8 8	オ. 3 4 5 6 7	カ. 3 4 5 6 7 8